Advances in the Bayesian Occupancy Filter framework using robust motion detection technique for dynamic environment monitoring

Qadeer Baig, Mathias Perrollaz, Christian Laugier

Advances in the Bayesian Occupancy Filter framework using robust motion detection technique for dynamic environment monitoring

Qadeer Baig, Mathias Perrollaz, and Christian Laugier,

***Abstract*—The Bayesian Occupancy Filter provides a framework for grid-based monitoring of the dynamic environment. It allows to estimate dynamic grids, containing both information of occupancy and velocity. Clustering such grids then provides detection of the objects in the observed scene.**
In this paper we present recent improvements in this framework. First, multiple layers from a laser scanner are fused using opinion pool, to deal with conflicting information. Then a fast motion detection technique based on laser data and odometer/IMU information is used to separate the dynamic environment from the static one. This technique instead of performing a complete SLAM (Simultaneous Localization and Mapping) solution, is based on transferring occupancy information between consecutive data grids, the objective is to avoid false positives (static objects) like other DATMO approaches. Finally, we show the integration with Bayesian Occupancy Filter (BOF) and with the subsequent tracking module called Fast Clustering-Tracking Algorithm (FCTA). We especially show the improvements achieved in tracking results after this integration, for an intelligent vehicle application.

***Index Terms*—Baysian Occupancy Filter, Motion detection, Multi layer laser data fusion, Moving objects tracking.**

I. Introduction

IN the field of Advanced Driver Assistance Systems (ADAS), many current approaches rely on the perception of the road scene. Particularly, the detection and tracking of the objects in the scene is essential for prediction of risky driving situations. Among recent approaches for risk estimation, the authors in [1] proposed to model and recognize the behavior of road scene participants in order to refine the prediction of their future trajectories. It allows a long term prediction, compared to classical geometric approaches. This is very promising for long term risk assessment, but reasoning on behaviors requires to separate environment into static and dynamic parts.

A huge amount of work has been done to detect moving objects, especially by the vision community [2]–[5]. These techniques have primarily been based on background subtraction or motion cues. Similar techniques have also been used in occupancy grids to detect moving objects [6], [7]. These techniques are based on inconsistencies observed for new data by comparing them with maps constructed by SLAM. [8] and [9] have also proposed model-based techniques to detect moving objects on the roads in the context of autonomous vehicle driving. More recently, Dempster-Shafer theory based grids (evidential grids) have been used to detect moving objects using conflict analysis techniques [10], [11]. However finding appropriate values of the operators involved is usually a challenging task. The technique described in [12] uses a modified concept of particle filers to monitor dynamic environment.
There exist various approaches for such classification of the environment. Having a very precise map of the environment, coupled with a very accurate localization algorithm, can be costly. Thus to perform a SLAM-based localization would be more realistic for commercial vehicles, the computed occupancy grid [13] providing a description of the static environment. More elaborated techniques like [7] or [8] improve this approach by performing detection and classification simultaneously, in a model-based-tracking framework.

When considering more generally the problem of Detection and Tracking of Moving Objects (DATMO), a lot of different methods have been proposed in the literature. In [14], Petrovskaya *et al.* propose to classify these methods in three categories: *Traditional DATMO*, *Model-based DATMO* and *Grid-based DATMO*. In this paper we are interested in using *Grid-based DATMO*. It is based on the use of the Bayesian Occupancy Filter (BOF), originally developed in [22] and then expanded in [15], [16], combined with Fast Clustering and Tracking Algorithm (FCTA) [17] for monitoring the dynamic environment. An important work to cite here is [18]. The idea was inspired by the original BOF formulation [22], but integrates also prior map knowledge. However this new formulation has comparable results as that of the original BOF framework but it lacks the static-dynamic division of the cells.

Although the BOF and FCTA combination works fine for properly tuned FCTA parameters, finding appropriate values for these parameters can be a time consuming task. Without proper parameter values FCTA may miss true tracks (for bigger parameter values) or may give false tracks (for smaller parameter values). This dependency of FCTA on parameters is necessary because BOF estimates the occupancy and velocity values for both static and dynamic parts of the environment. BOF framework performs the environment monitoring even when no motion sensor is available. Since objective of FCTA is to cluster and track the moving objects, it is advantageous

Q. Baig is with department of computer science Comsats Institute of Information and Technology Abbottabad, Pakistan, QadeerBaig@ciit.net.pk.
M. Perrollaz and C. Laugier are with E-Motion team of Inria Rhône Alpes, 38334 Montbonnot Saint Martin, France, Email: First.Last@inria.fr.

to remove the static parts from the output of BOF and input FCTA only detected moving parts. If input of FCTA only consists of likely moving parts of the environment, the dependence of FCTA's performance on the values of its parameters can be relaxed.

In this work we first present our new approach for building occupancy grids from multiple layers of laser scans. Then we propose a fast and efficient method for static/dynamic environment classification from the occupancy grid generated in the last time-step and show its integration within the BOF framework, to provide a more accurate representation of cell velocities. BOF also takes the occupancy grid generated by the fusion module as an input, and outputs an occupancy and velocity grid that is used by the FCTA. Therefore, we also present the integration of our approach with FCTA. This integration has enabled us to remove a large number of false tracks. Our approach is different from other grid based methods in the sense that usually a complete SLAM solution is implemented to separate the moving parts from the static parts (as in [9]). In current work we have developed a technique that deals with only few consecutive frames to detect moving parts rapidly, and can be easily implemented in real time.

The paper is organized as follow: Section II briefly describes the Bayesian Occupancy Filter framework. Along with reminders of the basic approach, it presents the method used to build occupancy grids from multiple layers of laser scans. It also introduces the subsequent grid clustering technique (FCTA). Next in section III we detail our technique to detect moving objects from the sensor data. Section IV details the integration of this motion detection module within the BOF framework. This approach is tested for an intelligent vehicle application. Section V describes our demonstrator vehicle used to get data sets for this work, and presents some results. The paper concludes with future perspectives in section VI.

II. The Bayesian Occupancy Filter (BOF) Framework

In this section we briefly introduce the BOF framework with the FCTA module to detect and track the moving objects. In this framework, the idea is to perform sensor fusion and environment monitoring at low level, hence the fast and efficient grid-based representation used during all the process. Data association is delayed in order to avoid association-related problems during the early stages. The objects are only retrieved at the end of the processing through clustering of the dynamic grid. To summarize, the complete processing from sensor data to objects can be divided into three stages: i) Multi sensor fusion using occupancy grid representation ii) filtering and estimation of dynamic grid and finally iii) clustering of the dynamic grid and tracking.

A. Sensor fusion from multiple lidar layers

Consider multiple layers of laser scans. They could be provided either by a multiple-layers laser scanner or by multiple single-layer laser scanners. Each layer can be used to compute an occupancy grid using the classical range finder probabilistic sensor model described in [19]. In order to retrieve a single grid for representation of the environment, the data from all these layers are merged using the approach described in [20]. This approach fuses the sensory information by using Linear Opinion Pools [21].

The principle is to generate a posterior distribution over the occupancy of a cell C of the grid given the opinion of m sensors $\{Y_1 \dots Y_m\}$. For each cell, each sensor gives two quantities: its estimation for the occupancy of the cell $P(C|Y_i)$ and $w_i(C)$, a measure of the confidence for such estimations. The idea is to shut-down those sensors that do not give relevant information to the process by assigning a low weight to them. The fusion of all sensory information will be as follows:

$$P(C|Y_1 \dots Y_m) = \alpha \sum_{i=1}^{m} w_i(C) P(C|Y_i) \tag{1}$$

where $\alpha = \left[\sum_{i=1}^{m} w_i(C) \right]^{-1}$ is a normalization factor for the weights. Equation (1) is used to generate 2D-occupancy grids. For each sensor Y_i we must define $P(C|Y_i)$, the probability of a cell being occupied given the sensor information; and $w_i(C)$, the confidence on the opinion. The confidence includes two components : one related to the geometry (e.g. if the beam is close to the ground), the second one is related to the probability of the cell to be actually visible by the sensor (details in [20]). Note that we assume independence among cells, in order to be efficient in computing equation (1), for each cell in parallel.

There are two advantage with this method. First, the confidence on occupancy is computed independently for each cell of the grid. Therefore, if a layer hits the road surface, the confidence will be low and it will not interfere with other layers. Second, it has the advantage of reducing the errors due to conflicting information from the multiple layers. For example, if a layer hits an object (meaning that the corresponding cell is occupied), and an upper layer goes over the object (meaning that the cell is free), both information are in conflict. As seen on Fig. 1, standard Bayesian fusion would lead to low occupancy probability, while the proposed approach would see the cell as probably occupied.

B. Filtering the grid using the Bayesian Occupancy Filter

The Bayesian Occupancy Filter (BOF) provides filtering capability, as well as the ability to estimate a velocity distribution for each cell of a grid. It is an adaptation of the Bayesian filtering methodology to the occupancy grid framework, based on a prediction-estimation paradigm, as shown in figure 2. As an input, it uses an observed occupancy grid. On its output, it provides an estimated occupancy grid as well as a set of

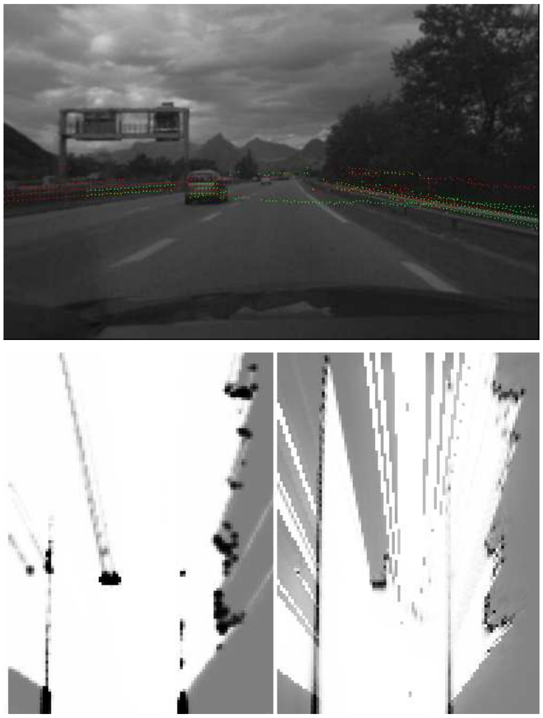

Fig. 1. (up) Left image from the stereo camera. (down) Occupancy grids computed from multiple layers of laser scans: (left) with Bayesian Fusion, (right) with our new approach. Due to conflicting information, the left fence disappears with Bayesian fusion, while still being visible with our method.

velocity grids, representing the probability distribution over possible velocities for each cell.

In this context, the prediction step propagates cell occupancy and velocity distributions of each cell in the grid and obtains the prediction $P(O_c^t \ A_c^t)$ where $P(O_c^t)$ denotes the occupancy distribution and $P(A_c^t)$ denotes the antecedent (velocity) distribution of a cell c at time t (velocity of a cell depends on its previous position called antecedent, if we know the previous position of a cell we can calculate its velocity because size of cells is fixed and we know the time interval as well). In the estimation step, $P(O_c^t \ A_c^t)$ is updated by taking into account the observations yielded by the sensors $\prod_{i=1}^{S} P(Z_i^t \mid A_c^t \, O_c^t)$ to obtain the posterior state estimate $P(O_c^t \ A_c^t \mid [Z_1^t \cdots Z_S^t])$ where Z_i^t denotes the observation of sensor i at t. This allows us to compute by marginalization $P(O_c^t \mid [Z_1^t \cdots Z_S^t])$ and $P(A_c^t \mid [Z_1^t \cdots Z_S^t])$, which will be used for prediction in the next iteration.

C. Detecting objects with the Fast Clustering and Tracking Algorithm

While the dynamic grid representation provided by the BOF can be sufficient for some applications (e.g. navigation), moving object detection requires to retrieve an object level representation of the scene. This cannot be directly reached from the occupancy grid, and therefore a clustering algorithm is necessary. An algorithm adapted to the BOF framework is the “Fast Clustering Tracking Algorithm” described in [17]. It has the major interest to create clusters considering not only the connectivity in the occupancy grid, but also the

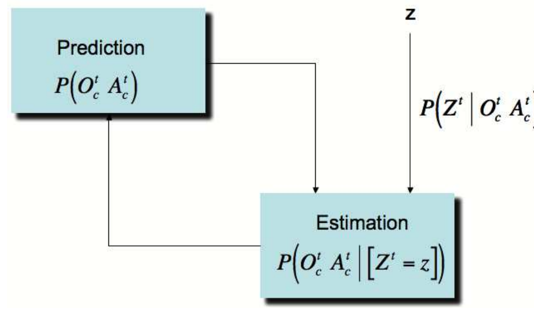

Fig. 2. Bayesian filtering in the estimation of occupancy and velocity distributions in the BOF grid [22].

Mahalanobis distance between cells in the estimated velocity grids. Thus two connected cells with different velocities are not merged during the clustering process. This is very important because two closely located such clusters may actually belong to two different objects moving with different velocities. Thus addressing the partial occlusion problem elegantly.
FCTA includes a Kalman filter for target tracking and a ROI prediction approach that allows computation to be performed in real time. The output of the algorithm is a set of tracked objects, with position, velocity and associated uncertainties.

The integration of this detection and tracking algorithm with the BOF algorithm aims at providing a single application that takes its input from the installed sensors and outputs the tracks to be used for risk assessment. Like other approaches FCTA also suffers from false tracks detection problem, so a pre-classification of cells into static and dynamic categories can help in this regard. The *motion detection* module capable of classifying the environment into static and dynamic regions can serve to ease this problem.

III. Motion Detection

A. Outline of the approach

In this section we detail the technique that we have developed to find moving parts of the environment. This motion detection module is situated in the processing chain just before the BOF. The input to this module consists of an occupancy grid generated by the fusion module, described in previous section. Let us represent this occupancy grid at time t as OG_t and an ith cell in this grid can be accessed as $OG_t[i]$ where $0 \leq i < N$ with N being the total number of cells of this occupancy grid. The value of each cell of this grid is between 0 and 1 i.e. $0 \leq OG_t[i] \leq 1$ and represents internal belief of the ego vehicle about the occupancy state of each cell. 0 means empty and 1 means occupied.

The module also requires proprioceptive information from the vehicle. Here we consider using an Inertial Measurement Unit (IMU), which provides at time t -along with other information- two components of velocity $v_t = (v_x, v_y)$ and values of quaternion components for orientation $Q_t =$

(q_0, q_1, q_2, q_3). From these information we calculate the translational and rotational velocities $u_t = (\nu_t, \omega_t)$ of the vehicle as follows:

$$\nu_t = \sqrt{v_x^2 + v_y^2} \quad (2)$$

To compute rotational velocity of the vehicle we calculate yaw angle of the vehicle from the quaternion as follows:

$$\mathcal{Y} = atan2(2*(q0*q3+q1*q2), 1-2*(q2*q2+q3*q3)) \quad (3)$$

And if dt is the time difference between two successive data frames at time t and $t-1$, then rotational speed ω at time t is equal to the yaw rate given as:

$$\omega_t = \frac{\mathcal{Y}_t - \mathcal{Y}_{t-1}}{dt} \quad (4)$$

B. Motion Detection Algorithm

At each time t, the algorithm that consists of following steps :

Step I) Free and occupied counts arrays:
For each new input occupancy grid OG_t we create two count arrays having same dimensions as that of OG_t, the first one called $FreeCount_t$ and the other one called $OccupiedCount_t$ to keep count of the number of times a cell has been observed "free" and number of times it has been observed "occupied" respectively. These arrays are initialized from OG_t as follows:

$$OccupiedCount_t[i] = \begin{cases} 1, & \text{if } OG_t[i] > 0.5 \\ 0, & \text{otherwise} \end{cases} \quad (5)$$

and

$$FreeCount_t[i] = \begin{cases} 1, & \text{if } OG_t[i] < 0.5 \\ 0, & \text{otherwise} \end{cases} \quad (6)$$

Step II) Counts update from previous step:
Suppose $FreeCount_{t-1}$ and $OccupiedCount_{t-1}$ are the updated count arrays at time $t-1$. We want to update new counts $FreeCount_t$ and $OccupiedCount_t$. Since vehicle has undergone a position change determined by $u_t = (\nu_t, \omega_t)$, there is no direct correspondence between cells of two occupancy grids OG_t and OG_{t-1}. We must find the transformation that maps a cell in OG_{t-1} to a cell in OG_t using u_t. This situation is shown in figure 3, OG_{t-1} has origin at O_{t-1} and OG_t has origin at O_t. To find the transformation suppose $O_{t-1} = (x_{t-1}, y_{t-1}, \theta_{t-1}) = (0, 0, 0)$ is the pose (position and orientation) of the occupancy grid at time $t-1$ (i.e of OG_{t-1}) and we want to find $O_t = (x_t, y_t, \theta_t)$, the pose of OG_t under u_t. Considering a circular motion trajectory (which means that the object with both linear and rotational velocity is supposed to actually move along the circular path. For short periods of time its is an acceptable approximation.), the pose of O_t w.r.t O_{t-1} is given as:

$$\begin{bmatrix} x_t \\ y_t \\ \theta_t \end{bmatrix} = \begin{bmatrix} \nu_t/\omega_t * sin(\omega_t * dt) \\ \nu_t/\omega_t - \nu_t/\omega_t * cos(\omega_t * dt) \\ \omega_t * dt \end{bmatrix} \quad (7)$$

Although it is a good approximation for a relatively accurate motion sensor, localization errors may occur due to sensor

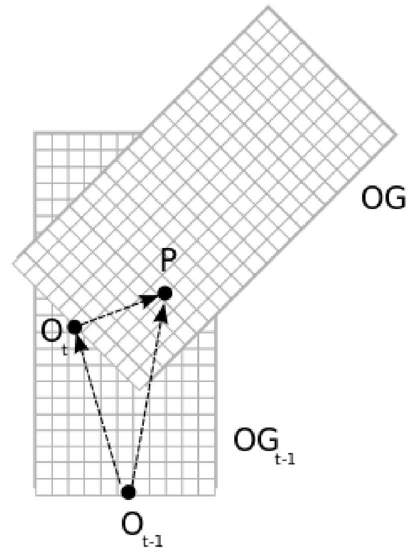

Fig. 3. Position of the grid at time instants $t-1$ and t. Vehicle undergoes a motion of $u_t = (\nu_t, \omega_t)$ to move from O_{t-1} to O_t. We need to find the position of point P of grid OG_{t-1} in grid OG_t.

uncertainty and external factors (like vehicle slipping). To fix these problems we use a pose correction measure similar to the one given in [19]. An uncertain region is defined around the new position at time t (given by equation 7) calculated using circular motion model. Different pose samples are taken from this region and for each pose sample a score function is evaluated using new data OG_t and free and occupied counter values. The function is given below (OC for Occupied Counter, FC for Free Counter, and N for total grid cells):

$$score = \sum_j^N \begin{cases} 1, \quad \text{if } (OG_t[j] > 0.5) \text{ and} \\ (OccupiedCount_{t-1}[i] > FreeCount_{t-1}[i]) \\ \\ -1, \quad \text{if } (OG_t[j] < 0.5) \text{ and} \\ (OccupiedCount_{t-1}[i] > FreeCount_{t-1}[i]) \\ \\ 0, \quad \text{otherwise} \end{cases} \quad (8)$$

(Note that $OG_t[j] = 0.5$ gives no information whether the cell is occupied or empty.) Here i and j are corresponding cells in occupancy grids at $t-1$ and t respectively, as explained below. The pose sample giving the highest value of this score is taken as the correct current pose of the vehicle improving the overall motion detection results.

An important thing to note here is that we are concerned with the localization of few consecutive frames only. We do not solve the complete SLAM problem, making the technique very fast. The pose estimation between two frames may not be exact but the error does not accumulate over longer periods of time because the map is very local, restricted to one small window of one data frame. Ego vehicle is always located on the middle point of the bottom edge.

To map a cell of grid OG_{t-1} to grid OG_t we proceed as follows. Suppose point P (shown in figure 3) is the center of a cell in grid OG_{t-1} and we want to find its corresponding cell in grid OG_t. We define following two pose manipulation operations:

Combination: If P_{ij} is the pose of origin j w.r.t origin i and $P_{jk} = [x_{jk}, y_{jk}, \theta_{jk}]^T$ is the pose of origin k w.r.t j then the pose of k w.r.t i denoted as $P_{ik} = [x_{ik}, y_{ik}, \theta_{ik}]^T$ is given as:

$$P_{ik} \equiv \oplus(P_{ij}, P_{jk}) = \begin{bmatrix} x_{jk}\cos(\theta_{ij}) - y_{jk}\sin(\theta_{ij}) + x_{ij} \\ x_{jk}\sin(\theta_{ij}) + y_{jk}\cos(\theta_{ij}) + y_{ij} \\ \theta_{ij} + \theta_{jk} \end{bmatrix} \tag{9}$$

Inverse: For the pose P_{ij} the reverse pose relationship $P_{ji} = [x_{ji}, y_{ji}, \theta_{ji}]^T$ (pose of i w.r.t j) is defined as:

$$P_{ji} \equiv \ominus(P_{ij}) = \begin{bmatrix} -x_{ij}\cos(\theta_{ij}) - y_{ij}\sin(\theta_{ij}) \\ x_{ij}\sin(\theta_{ij}) - y_{ij}\cos(\theta_{ij}) \\ -\theta_{ij} \end{bmatrix} \tag{10}$$

Since the pose of O_t w.r.t O_{t-1} is $P_{O_{t-1}O_t} = [x_t, y_t, \theta_t]^T$ and point P has pose $P_{O_{t-1}P} = [x, y, 0]^T$ w.r.t O_{t-1}. The pose of this point P w.r.t O_t is calculated as.

$$P_{O_tP} = \oplus(P_{O_tO_{t-1}}, P_{O_{t-1}P}) \tag{11}$$

or

$$P_{O_tP} = \oplus(\ominus(P_{O_{t-1}O_t}), P_{O_{t-1}P}) \tag{12}$$

First two components of P_{O_tP} give x and y position of point P w.r.t origin O_t. From these x and y values we can easily calculate the index of the cell where point P lies in grid OG_t. These transformations will map a cell having index i in OG_{t-1} to a cell having index j in grid OG_t. If this cell j is visible in OG_t i.e $0 \leq j < N$ then we can update new count values for this cell as follows:

$$FreeCount_t[j] = FreeCount_t[j] + FreeCount_{t-1}[i] \tag{13}$$

and ($OCount$ in the following equations is $OccupiedCount$ to fit the equation in a line)

$$OCount_t[j] = OCount_t[j] + OCount_{t-1}[i] \tag{14}$$

We repeat this process for all cells of grid OG_{t-1} to update counts values in grid OG_t.

Step III) Motion detection:

After the counts arrays have been updated as explained above the motion grid can be calculated from the new data using following heuristic:

$$MotionGrid_t[i] = \begin{cases} 1, & OG_t[i] > 0.5 \text{ and} \\ & FreeCount_t[i] > m * OCount_t[i] \\ 0, & \text{otherwise} \end{cases} \tag{15}$$

Here m is a sensor specific input parameter. Its empirically learnt value in our experiments is 2. So a cell has a moving object in it if it is occupied at current time instant and previously it was observed as free for sufficient number of times (at least twice the number of times it has been observed as occupied). After this processing, $MotionGrid_t$ has 1 in the cells which are detected as belonging to moving objects.

The final measure that we have taken to remove false positives is the implementation of a road border detection (as detailed in [23]) technique. In this technique road borders are searched in vertical directions of current vehicle orientation using a window based approach. The window giving a score greater than a certain threshold signals the presence of a road border. When road borders are detected with high confidence, moving cells detected beyond the road border are ignored.

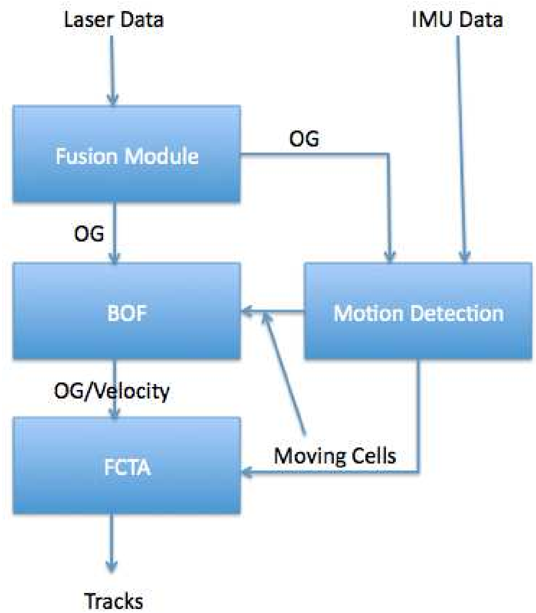

Fig. 4. Block diagram of the system.

IV. Integration with the BOF framework

A. Integration with BOF

We have updated the BOF implementation to take into account the motion detection results. The motion grid is used as an input for updating the BOF. If the input motion grid tells that a cell belongs to a static object, then during prediction and update cycles of BOF the cell's velocity distribution over the velocity range is set to uniform for all discrete velocity values. This essentially means that no velocity information for a given cell is available and the cell is labeled as static in the current BOF implementation. However, if the cell has been detected as belonging to a moving object, then the velocity distribution prediction and the update cycle are carried out normally. In formal terms as outlined in [17], this change in the parametric form of dynamic model can be stated as:

$$P(A_i^t | A_i^{t-1}) = \begin{cases} (1-\epsilon)P(A^{t-1}_{A_i^{t-1}}) + \epsilon/\|A_i\| \\ \quad \text{if } MotionGrid_t[i] > 0 \\ \\ 1/\|A_i\| \quad \text{otherwise} \end{cases}$$

where A_i^t is the set of antecedents of cell i at time t and ϵ is a parameter of BOF, modeling the prediction error probability.

The antecedent of a cell i at time t means the cell j at time $t-1$ where the same object (currently seen in cell i) was seen. This knowledge along with the knowledge of time difference and cell sizes lets us calculate the object velocity. Complete derivation of this equation can be found in [17].

B. Integration with FCTA

We have also updated the FCTA implementation to take into account the motion detection results. The cells which do not possess the velocity information are now ignored during the clustering step. While generally most of the areas belong to static objects and are detected as static by the motion detection module, two main advantages are expected from this strategy: (i) the clustering stage of the algorithm is highly accelerated by the reduction of hypotheses, and (ii) the wrong moving clusters are ignored because they are not considered for clustering, even with the relaxed FCTA parameters. A block diagram of the system is shown in figure 4.

V. Experimentation

A. Experimental Platform

Our experimental platform is a Lexus LS600h car, shown in Fig. 5. The car is equipped with following sensors (Fig. 5 and 6): two IBEO Lux lidars placed in the front bumper, a TYZX stereo camera situated behind the windshield, and an Xsens MTi-G sensor (IMU with GPS). Extrinsic calibration of the sensors is done manually for this work. Note that, thanks to the grid-based approach and considering the resolution of the grid, a slight calibration error has very little impact on the final results.

Fig. 5. Lexus LS600h car equipped with two IBEO Lux lidars and a TYZX stereo camera

The hardware specification are the following: IBEO Lux LIDAR laser scanner provides four layers of up to 200 beams with a sampling period of 40 ms. The angular range of each lidar is $100°$, and the angular resolution is $0.5°$. The on-board computer is equipped with 8GB of RAM, an Intel Xeon 3.4GHz processor and a NVIDIA GeForce GTX 480 for GPU. The observed region is 60 m long by 20 m wide, with a maximum height of 2 m (due to different vertical angle of the 4 layers of each laser scanner). Cell size for the occupancy grids is 0.2m x 0.2 m. Xsens MTi-G is deployed in the middle of the rear wheel axis. In this work we do not use the stereo vision camera.

Fig. 6. (left) the MTi-G XSens IMU unit, (right) the trunk of the car, including the PC used for calculation.

B. Results

1) Motion detection: Some results of the motion detection module are shown in figures 7 and 8, (rectangles around the objects are drawn manually to highlight them). As expected, the moving objects are properly detected. For example, Fig. 7 shows correct motion detection in a scenario with two cars. The car moving around a roundabout in Fig. 8 has also been successfully detected. Some noise is also visible on the results, mainly due to two causes: first, localization errors along with the circular motion model may result in some errors in the estimation of the motion; second, the decision function is too rough for taking correct decisions in every situation (especially for the regions having non fixed objects like bushes and grass). The results would benefit from replacing this function by a probabilistic model.

2) Integration with FCTA: The following statistics, with a dataset of about 13 minutes of driving on open roads, give an insight into the improvements gained with our new implementation. When the motion detection module is not used, 22303 moving objects (not tracks) in total are detected. The activation of the motion detection module with all other parameters being equal detects 4796 moving objects in all frames in total. This example shows the advantage of the motion detection module because it allows us to remove most of the false moving objects while leaving most of the true dynamic objects. When considering tracks and not only detections, the results are also promising, as shown on table I.

Some qualitative FCTA tracking results with and without motion detection module activated (with all other parameters being same) are also shown in Fig. 9. Red rectangles are the detected tracks by FCTA in the shown scenario. We clearly see that most of the false positives have been removed.

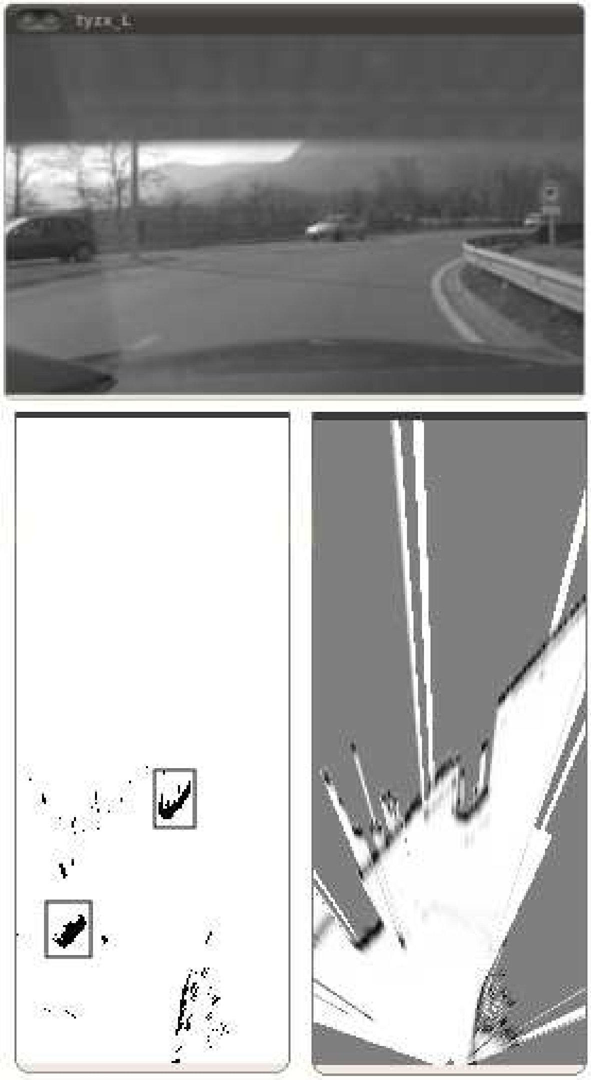

Fig. 7. Motion detection results of two cars. Top, scenario, bottom right input fused grid, bottom left resulting motion grid. The noise on the bottom right is due to bushes along the road.

TABLE I
STATISTICS OF DETECTED TRACKS WITHOUT AND WITH MOTION DETECTION (MD) MODULE ACTIVATED.

Data Set	Duration	Tracks without MD	Tracks with MD	True tracks
1	13:00	279	56	37
2	9:30	223	48	29
3	7:49	217	43	31

3) Computation time: All the components of the method are designed to be implemented on GPU with highly parallel processing. The measured average computation time for each module is:

- Multi-layer fusion (GPU): 8ms
- BOF update (GPU): 10ms
- Motion detection (CPU): 2.6 ms
- FCTA (CPU): 0.34 ms

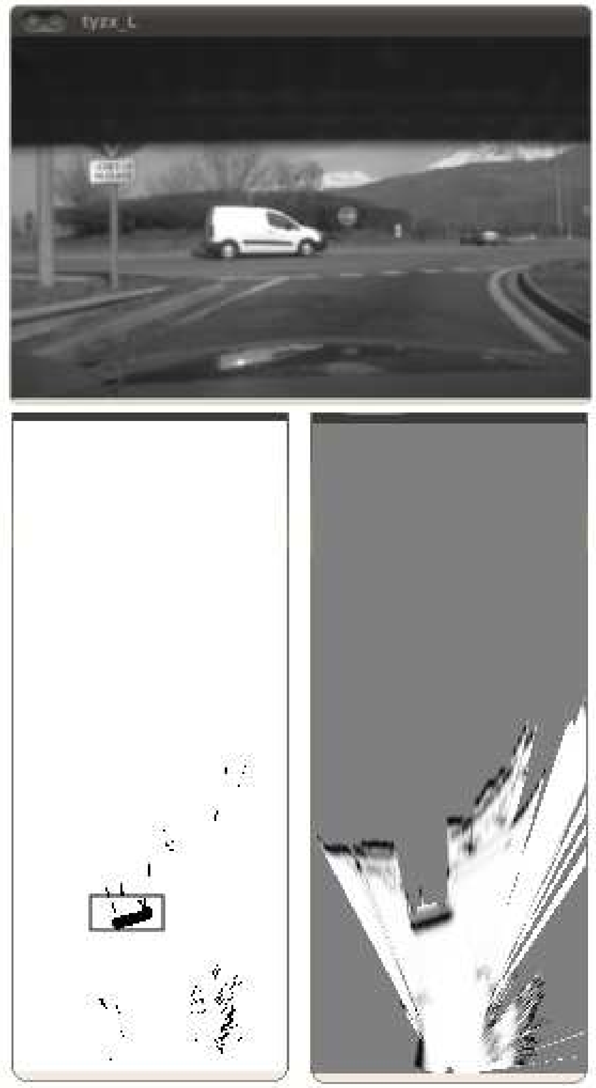

Fig. 8. Motion detection results of a car on a roundabout.

Thanks to the GPU implementation, the two first stages of the approach can work in real time. As expected, the motion detection module is fast despite the CPU implementation. We hope to obtain an important gain from the GPU implementation. Moreover, this module allows to reduce significantly the computation time of the FCTA, by reducing the number of tracking hypotheses. As an example, for the same scene processed without and with the motion detection module, the FCTA computation time is : 0.5 and 0.34 ms respectively.

VI. CONCLUSION AND FUTURE WORK

In this paper we have presented our current implementation of the Bayesian Occupancy Filter framework for dynamic environment monitoring. Compared to the previous implementation, we added two modules which are well suited for the intelligent vehicle application. First, we propose a new approach for fusing the information from multiple

Fig. 9. Example tracking results.For each scenario : Left, FCTA results without motion detection module activated; right, same scenario with motion detection module activated.

layers of laser scanners, and computing the corresponding occupancy grid. Then, a fast technique to find moving objects from this grid data is proposed. The presented technique does not require to perform complete SLAM to detected moving objects but uses grid data along with odometer/IMU information to transfer occupancy information between two consecutive grids. We have also presented its integration with Bayesian Occupancy Filter (BOF) and Fast Clustering-Tracking Algorithm (FCTA). We have seen that after this integration we were able to remove a significant part of the detections resulting from the static environment.

In the future, we plan to change the rather ad hoc decision module that is currently based on occupied and free counter values to a more formal probabilistic function that also takes into account the uncertainty effects on the neighboring cells to accommodate the localization errors. We also plan to implement multiple motion models, to track moving objects with more accuracy and over longer periods of time.

Acknowledgment

This work was jointly supported by Inria Grenoble Rhône-Alpes, Toyota Motor Europe, and the project ”Prefect” from the IRT ”Nano Elec” (A French Research and Innovation Project operated by the CEA and supported by the French Research Agency ANR). We would like to express our special thanks to Dr. Dizan Vasquez and Dr. Trung Dung Vu for their technical supports and fruitful suggestions.

References

[1] C. Laugier, I. Paromtchik, M. Perrollaz, M. Yong, J. Yoder, C. Tay, K. Mekhnacha, and A. Negre, “Probabilistic analysis of dynamic scenes and collision risks assessment to improve driving safety,” in *Intelligent Transportation Systems Magazine, IEEE*, vol. 3, no. 4, pp. 4 –19, 2011.

[2] R. Jain, W. N. Martin, and J. K. Aggarwal, “Segmentation through the detection of changes due to motion,” in *Computer Graphics and Image Processing*, vol. 11, no. 1, pp. 13–34, September 1979.

[3] Z. Zhang, “Iterative point matching for registration of free-form curves and surfaces,” in *International Journal of Computer Vision*, vol. 13, no. 2, pp. 119–152, 1994.

[4] D. Li, “Moving objects detection by block comparison,” in *IEEE International Conference on Electronics, Circuits and Systems, 2000.*, vol. 1, pp. 341–344.

[5] S. Taleghani, S. Aslani, and S. Shiry, “Robust Moving Object Detection from a Moving Video Camera Using Neural Network and Kalman Filter”. Springer-Verlag, 2009, pp. 638–648. [Online]. Available: http://dl.acm.org/citation.cfm?id=1575210.1575268

[6] C.-C. Wang, C. Thorpe, and S. Thrun, “Online simultaneous localization and mapping with detection and tracking of moving objects: theory and results from a ground vehicle in crowded urban areas,” in *IEEE International Conference on Robotics and Automation*, Taipei, Taiwan, 2003, vol. 1, pp. 842–849.

[7] T.-D. Vu, J. Burlet, and O. Aycard, “Grid-based localization and local mapping with moving object detection and tracking,” in *Information Fusion*, vol. 12, no. 1, pp. 58 – 69, 2011, Special Issue on Intelligent Transportation Systems. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1566253510000187

[8] A. Petrovskaya and S. Thrun, “Model based vehicle detection and tracking for autonomous urban driving,” in *Autonomous Robots*, vol. 26, no. 2, pp. 123–139, 2009.

[9] T.-D. Vu and O. Aycard, “Lased-based detection and tracking moving object using data-driven markov chain monte carlo,” in *IEEE International Conference on Robotics and Automation*, Kobe, Japan, May 2009.

[10] J. Moras, V. Cherfaoui, and P. Bonnifait, “Moving objects detection by conflict analysis in evidential grids,” in *IEEE Intelligent Vehicles Symposium*, Baden-Baden, Allemagne, June 5-9, 2011, pp. 1120–1125. [Online]. Available: http://hal.archives-ouvertes.fr/hal-00615304/en/

[11] ——, “Credibilist occupancy grids for vehicle perception in dynamic environments,” in *IEEE International Conference on Robotics and Automation*, Shanghai, China, May 9-13 2011, pp. 84–89. [Online]. Available: http://hal.archives-ouvertes.fr/hal-00615303/en/

[12] R. Danescu, F. Oniga, and S. Nedevschi, “Modeling and tracking the driving environment with particle-based occupancy grid,” in *IEEE Transactions on Intelligent Transportation Systems*, vol. 12, no. 4, pp. 1331–1342, 2011.

[13] A. Elfes, “Occupancy grids: A stochastic spatial representation for active robot perception,” in *Proceedings of the Sixth Conference Annual Conference on Uncertainty in Artificial Intelligence (UAI-90)*. New York, NY: Elsevier Science, 1990, pp. 136–146.

[14] A. Petrovskaya, M. Perrollaz, L. Oliveira, L. Spinello, R. Triebel, A. Makris, J.-D. Yoder, C. Laugier, U. Nunes, and P. Bessiere, “Awareness of road scene participants for autonomous driving”, in *Handbook of Intelligent Vehicles*. Springer, 2011.

[15] C. Tay, K. Mekhnacha, C. Chen, M. Yguel, and C. Laugier, “An efficient formulation of the bayesian occupation filter for target tracking in dynamic environments,” in *International Journal Of Autonomous Vehicles*, vol. T6, no. 1-2, pp. 155–171, 2007.

[16] C. Coué, C. Pradalier, C. Laugier, T. Fraichard, and P. Bessiere, “Bayesian Occupancy Filtering for Multitarget Tracking: an Automotive Application,” in *International Journal of Robotics Research*, vol. 25, no. 1, pp. 19–30, Jan. 2006.

[17] K. Mekhnacha, Y. Mao, D. Raulo, and C. Laugier, “Bayesian occupancy filter based ”Fast Clustering-Tracking” algorithm”, in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Nice, France, 2008.

[18] T. Gindele, S. Brechtel, J. Schroder, and R. Dillmann, “Bayesian occupancy grid filter for dynamic environments using prior map knowledge,” in *IEEE Intelligent Vehicles Symposium*, Karlsruhe, Germany, 2009, pp. 669–676.

[19] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents series)*, The MIT Press, Aug. 2005. [Online]. Available: http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20&path=ASIN/0262201623

[20] J. D. Adarve, M. Perrollaz, A. Makris, and C. Laugier, “Computing Occupancy Grids from Multiple Sensors using Linear Opinion Pools”, in *IEEE International Conference on Robotics and Automation*, St Paul, Minnesota, USA, May 2012.

[21] M. H. DeGroot, “Reaching a consensus”, *Journal of the American Statistical Association*, vol. 69, no. 345, pp. pp. 118–121, 1974. [Online]. Available: http://www.jstor.org/stable/2285509

[22] C. Coué, “Modèle bayésien pour l’analyse multimodale d’environnements dynamiques et encombrés: Application à l’assitance à la conduite en milieu urbain,” Ph.D. dissertation, INPG, National Polytechnic Institute of Grenoble, 2003.

[23] Q. Baig and O. Aycard, “Improving moving objects tracking using road model for laser data,” in *IEEE Intelligent Vehicles Symposium*, Alcala de Henares, Spain, June 3-7 2012.

Dr. Qadeer Baig did his MS and PhD (2008 and 2012 respectively) from University of Grenoble 1, Grenoble France in the field of multi sensor data fusion for the detection and tracking of moving objects for intelligent transport systems. He also worked with E-motion team of Inria Rhône Alpes as post doctoral researcher. He is now affiliated with Comsats Institute of Information Technology Abbottabad Pakistan as Assistant Professor.

Dr. Mathias Perrollaz graduated from National Polytechnic Institute of Grenoble (INPG) in 2003, with a major in signal and image processing. He started working on ITS at the Images and Signals Laboratory in Grenoble (CNRS), and then attended the perception team of the LIVIC (INRETS). He received the Ph.D. degree from the university of Paris 6 (UPMC) in 2008, for his work on multi-sensor obstacle detection. Since April 2009, he is working at INRIA Grenoble on probabilistic methods for ITS. Between may and september 2011, he worked on perception for robotic manipulators at Ohio Northern University, USA. Mathias Perrollaz also taught in Paris-10 and Grenoble-2 Universities. His major research interests are computer-vision, ITS, and probabilistic robotics.

Dr. Christian Laugier received the PhD and the State Doctor degrees in Computer Science from Grenoble University (France) in 1976 and 1987 respectively. He is a First class Research Director at INRIA (http://www.inria.fr) and he is the Scientific Leader of the e-Motion team-project (http://emotion.inrialpes.fr) common to INRIA Rhône-Alpes and to the LIG Laboratory (http://www.liglab.fr/). From 2007 to 2011 he was Deputy Director of the LIG Laboratory involving about 500 peoples; he was also Deputy Director of the Computer Science and Artificial Intelligence Laboratory (LIFIA) from 1987 to 1992. Since 2009, he is also Scientific Program Manager for Asia & Oceania at the International Affairs Department of INRIA. His current research interests mainly lie in the areas of Motion Autonomy, Probabilistic Reasoning, Embedded Perception, and Intelligent Vehicles. He has co-edited several books in the field of Robotics, and several special issues of scientific journals such as IJRR, Advanced Robotics, JFR, or IEEE Trans on ITS. In 1997, he was awarded the IROS Nakamura Award for his contributions to the field of Intelligent Robots and Systems, and in 2012 he received the IEEE/RSJ Harashima award for Innovative Technologies for his Outstanding contributions to embedded perception and driving decision for intelligent vehicles.

Dr. Christian Laugier is member of several scientific committees such as the Steering/Advisory Committees of the IEEE/RSJ IROS, FSR, and ICARCV conferences. He is also co-Chair of the IEEE RAS Technical Committee on AGV & ITS. He has been General Chair, Program Chair or co-Chair of international conferences such as IEEE/RSJ IROS97, IROS02, IROS08, IROS10, IROS12, or FSR07. In addition to his research and teaching activities, he co-founded four start-up companies in the fields of Robotics, Computer Vision, Computer Graphics, and Bayesian Programming tools. He also served as Scientific Consultant for the ITMI, Aleph Technologies, and Probayes companies.